LAB 8

Objective: Create a single **nginx** Pod, create a service that selects this Pod, and access it from your browser. Verify that the endpoint has been populated and that DNS resolves the service name.

Services define a virtual IP address that directs traffic to Pods that match a label selector.

Let's create a Pod that runs **nginx**. The manifest is in the **nginx.yaml** file (retrieve it from *Course Resources*).

```
cat nginx.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  namespace: default
  labels:
    app: nginx
spec:
  containers:
  - image: nginx
    ports:
      - containerPort: 80
    imagePullPolicy: IfNotPresent
    name: nginx
```

Create it with **kubectl**


```
$ kubectl create -f nginx.yaml
```

The service is defined in another manifest, **nginx-svc.yaml** (retrieve it from *Course Resources*). The selector uses the label defined in the Pod manifest **app: nginx**. The port definition sets port 80 as the container port to reach the application.

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  ports:
    - port: 80
  type: NodePort
  selector:
    app: nginx
```

Once you create the service, you will see the IP of the matching Pod in the endpoints list:

```
$ kubectl create -f nginx-svc.yaml
$ kubectl get svc
NAME         CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
nginx        10.0.0.162   <nodes>       80/TCP    3s
$ kubectl get endpoints
NAME         ENDPOINTS         AGE
nginx        172.17.0.4:80     6s
```

Reach the service with your browser by typing **minikube service nginx**.

Finally, verify that DNS is working. Creating a *sleeping* **busybox** container, and *exec* into it to run **nslookup**:

```
$ kubectl create -f busybox.yaml
$ kubectl exec -ti busybox -- nslookup nginx
Server:    10.0.0.10
```



```
Address 1: 10.0.0.10 kube-dns.kube-system.svc.cluster.local

Name:      nginx
Address 1: 10.0.0.162 nginx.default.svc.cluster.local
```

In the example above, the IP 10.0.0.162 returned by the DNS lookup corresponds to the service IP address.